2009 IEEE International Conference on Robotics and Automation
Kobe International Conference Center
Kobe, Japan, May 12-17, 2009

Adaptive Triangular Mesh Generation of Self-configuring Robot Swarms

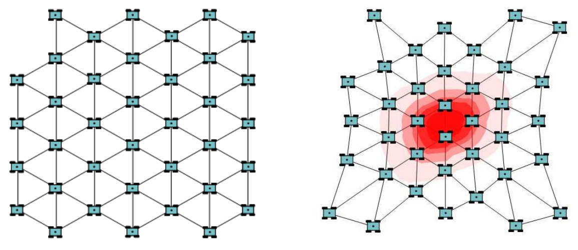
Geunho Lee[1], Nak Young Chong[1], and Henrik Christensen[2]

Abstract— **We address the problem of dispersing a large number of autonomous mobile robots toward building wireless *ad hoc* sensor networks performing environmental monitoring and control. For the purpose, we propose the adaptive triangular mesh generation algorithm that enables robots to generate triangular meshes of various sizes adapting to changing environmental conditions. A locally interacting, geometric technique allows robots to generate each triangular mesh with their two neighbor robots. Specifically, we have assumed that robots are not allowed to have the identifier, any pre-determined leaders or common coordinate systems, and any explicit communication. Under such minimal conditions, the positions of the robots were shown to converge to the desired distribution, which was mathematically proven and also verified through extensive simulations. Our preliminary results indicate that the proposed algorithm can be applied to the problem regarding the coverage of an area of interest by a swarm of mobile sensors.**

I. INTRODUCTION

With the advance of wireless and mobile networking technologies, much attention has been paid to the use of large-scale swarms of simple mobile robots for environmental or habitat monitoring. In particular, self-configuration of robot swarms requires a type of collective behavior that allows robots to disperse themselves in a certain area at a uniform spatial density. Thus, it is essential to properly coordinate the (relative) positions of robots, and this issue has been widely reported in the literature [2]-[12]. Taking steps to further improve those previous approaches, this work is aimed at presenting an algorithm that enables robot swarms to configure themselves adaptively in an area of interest with varying spatial densities. As illustrated in Fig. 1, robot swarms can explore an unknown area and detect and sense oil or chemical spills across the area. The contaminated area should be covered efficiently with mobile robots or sensors to investigate the degree and extent of contamination as quickly as possible and, if possible, prevent the possible expansion of the area. Therefore, in this paper, we address the problem of how to enable swarms of autonomous mobile robots to self-adjust their configuration or spatial density to fit local environmental conditions.

Based on our prior research on swarm configuration [16][17], we propose the adaptive triangular mesh generation algorithm that enables robot swarms to explore an area and adjust the interval between neighboring robots. The main objective is to provide robots with adaptive deployment capabilities to cover an area of interest more efficiently with variable triangular meshes according to sensed area conditions. This also can give us a more accurate picture of variations in the conditions of area. In this paper, the properties of the proposed algorithm are mathematically explained and the convergence is provided. We also demonstrated that a large-scale swarm of robots can establish a triangular mesh network adapting to varying degrees of contamination through extensive simulations. The results have been encouraging and indicate that self-configurable robot swarms are expected to be deployed for environmental or habitat monitoring.



(a) uniform deployment (b) adaptive deployment

Fig. 1. Uniform vs. adaptive triangular meshes of mobile robot swarms

II. BACKGROUND

Decentralized control for robot swarms can be broadly classified into global and local strategies according to whether sensors have range limits. Global strategies [2] may provide fast, accurate, and efficient deployment, but are technically non-feasible and lack scalability as the number of robots increases. On the other hand, local strategies are mainly based on interactions between individual robots inspired by nature. Local strategies can further be divided into biological emergence [3][4], behavior-based [5], and virtual physics-based [6]-[12] approaches. Many of the behavior-based and virtual physics-based approaches used such physical phenomena as electric charges [6], gravitational forces [7], spring forces [8][11][12], potential fields [9], van der Waals forces [10], and other virtual models.

Robot swarm configurations achieved by the above-mentioned local interactions may result in lattice-type networks. These configurations offer high level coverage and multiple redundant connections ensuring maximum reliability and flexibility from the standpoint of topology. Depending on whether there are interactions among all robots, the network can be classified into fully and partially connected

[1]G. Lee and N.Y. Chong are with the School of Information Science, Japan Advanced Institute of Science and Technology, 1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan (e-mail: {geun-lee, nakyoung}@jaist.ac.jp)

[2]H. Christensen is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA (e-mail: hic@cc.gatech.edu)

topologies [18]. The fully connected topologies have each robot interact with all of other robots simultaneously within a certain range. Thus, those approaches might over-constrain individual robots and frequently lead to deadlocks. On the contrary, using the partially connected topology, robots interact selectively with other robots, but are connected to all robots. For example, robots may choose to exert forces in a certain direction [11], where this selective interaction helps prevent them from being too tightly constrained. Due to similar reasons, robots are enabled to achieve faster formation without deadlocks [12].

In our earlier work [16], we presented self-configuration of a robot swarm that enables a large number of robots to configure themselves into a 2-dimensional plane with geographic constraints. A locally interacting geometric technique based on the partially connected topology provides a unique solution that allows robots to converge to the uniform distribution by forming an equilateral triangle with their two neighbors. By collecting such local behavior of each robot, a uniformly spaced swarm of robots was organized to fill in the environment. Compared with aforementioned works [3]-[12], our approach first is to construct uniformly spaced equilateral triangles conforming to the border of an unknown area when the robot sensors are subject to range and accuracy limitations. Secondly, an equilateral triangle lattice is built with a partially connected mesh topology. Among all the possible types of regular polygons, the equilateral triangle lattices can reduce the computational burden and become less influenced by other robots, due to the limited number of neighbors, and be highly scalable. Thirdly, the proposed local interaction is computationally efficient, since each robot utilizes only position information of other robots. Fourthly, our approach eliminates such major assumptions as robot identifiers, common coordinates, global orientation, and direct communication. More specifically, robots compute the target position without requiring memories of past actions or states, helping cope with transient errors.

For typical application examples, many works have been devoted to odor source localization. Jatmiko *et al.* [13] presented an algorithm for odor source localization in a changing environment based on particle swarm optimization. In [14], the authors introduced a combination of a local tracking controller and a high-level behavior-based framework that distribute robots into regions according to goal density. Krishnanand *et al.* [15] addressed the problem of multiple odor source localization using mobile robot swarms.

III. PROBLEM STATEMENT

We consider a swarm of mobile robots denoted as $r_1, \cdots, r_n$. It is assumed that all robots are within a swarm network configured by our previously proposed self-configuration method [16][17]. Each robot autonomously moves on a 2-dimensional plane. They have no leader and no identifiers, and do not share any common coordinate system, and do not retain any memory of past actions. Due to limited sensing range, they can detect the position of other robots only within a certain range. In addition, each robot does not communicate explicitly with other robots.

Let us consider a robot r_i with its local coordinates $\vec{r}_{x,i}$ and $\vec{r}_{y,i}$. Here, $\vec{r}_{y,i}$ defines the horizontal axis of r_i's coordinate system as its heading direction. It is straightforward to decide the vertical axis $\vec{r}_{x,i}$ by rotating the horizontal axis 90 degrees counterclockwise. The position of r_i is denoted as p_i. Note that p_i is $(0, 0)$ with respect to r_i's local coordinates. The distance between p_i and p_j is denoted as $dist(p_i, p_j)$. We define a uniform interval d_u, the desired distance between r_i and r_j. r_i detects the position $\{p_1, p_2, \cdots\}$ of other robots located within its sensing boundary *SB*, yielding a set of the positions O_i with respect to its local coordinates. Next, r_i can select two robots r_{s1} and r_{s2} within r_i's *SB* that we call the neighbors of r_i and denote their positions, $\{p_{s1}, p_{s2}\}$, as N_i. Given p_i and N_i, the *triangular configuration*, denoted by $\mathbb{T}_i$, is defined as a set of three distinct positions $\{p_i, p_{s1}, p_{s2}\}$.

As mentioned above, if robots detect an event such as oil or chemical spills within an area, they attempt to cooperate with each other to cover the area as efficiently as possible. The gradient in density across the area of contamination forces robots to adapt the interval between neighboring robots. For a certain point p_i occupied by r_i, the densities of contamination are expressed by k_i ranging between $0 \leq k_i \leq 1$, where $k_i = 0$ represents the maximum density and $k_i = 1$ corresponds to zero density.

Now, we formally address the ADAPTIVE TRIANGULAR MESH GENERATION PROBLEM as follows.

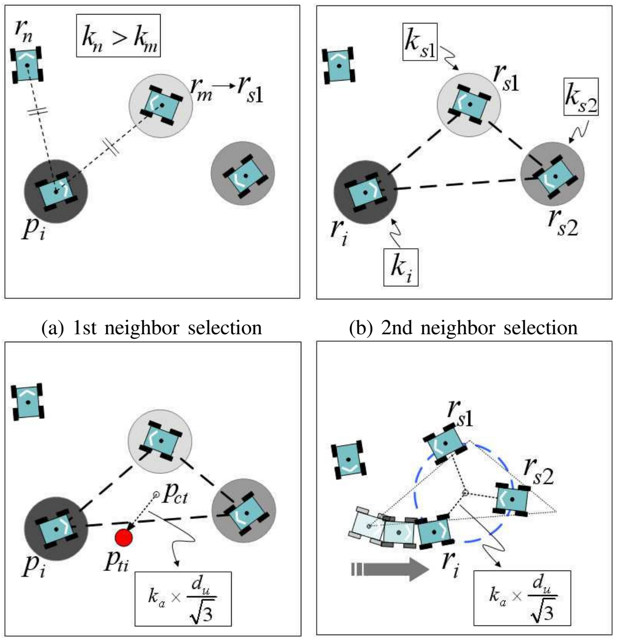
Given a swarm of mobile robots self-configured in a 2-dimensional plane, how to enable the robots to form triangular mesh patterns of various sizes adapting to varying environmental conditions.

Our solution approach to the above problem enables robots to disperse themselves into equilateral triangular patterns of various sizes according to the change in the degree of contamination within an area of interest. We will take advantage of the fact that, among all the possible types of *n*-polygons, the triangular mesh is highly scalable, and less influenced by the number of neighboring robots.

IV. ADAPTIVE TRIANGULAR MESH GENERATION ALGORITHM

A. Algorithm Description

At each time, r_i observes other robots within O_i to select a neighbor r_{s1} located the shortest distance. When there exist more than two candidates as illustrated in Fig. 2-(a), r_{s1} is determined according to the degree of contamination density k_m. Next, as illustrated in Fig. 2-(b), the second neighbor r_{s2} within O_i is selected such that the total distance from the position p_{s1} of r_{s1} to p_i passing through p_{s2} is minimized. Likewise, if there are more than two candidates for r_{s2}, r_i selects r_{s2} with a higher k_n. Then, r_i measures the angle ϕ between the line $\overline{p_{s1}p_{s2}}$ connecting two neighbors and the horizontal axis of the observing r_i's coordinate system. Next, as illustrated in Fig. 2-(c), the centroid p_{ct} in $\mathbb{T}_i$ ($\triangle p_i p_{s1} p_{s2}$)



(a) 1st neighbor selection (b) 2nd neighbor selection (c) target computation (d) moving to the target

Fig. 2. Illustrating how to adapt a triangular mesh to varying densities

is computed. Moreover, based on k_i, k_{s1}, and k_{s2} at each positions occupied by r_i, r_{s1}, and r_{s2}, r_i finds the average of densities k_a through the computation of $(k_i+k_{s1}+k_{s2})/3$ where k_{s1} and k_{s2} indicate the densities of r_{s1} and r_{s2}, respectively. Then, from p_{ct}, r_i executes a desired interval computation as follows: $d_a = k_a \times d_u/\sqrt{3}$. Utilizing d_a and ϕ, r_i calculates its target point $p_{ti} = (p_{ti,x}, p_{ti,y})$ located on a line with the previously calculated interval from p_{ct} and perpendicular to $\overline{p_{s1}p_{s2}}$ by the following equations: $(p_{ct,x} + k_a d_u \cos(\phi + \pi/2)/\sqrt{3},\ p_{ct,y} + k_a d_u \sin(\phi + \pi/2)/\sqrt{3})$. Finally, as presented in Fig. 2-(d), r_i moves toward p_{ti}. By doing this process repeatedly, r_i can form a triangular lattice.

B. Algorithm Property

Let's consider a triangle whose centroid is p_{ct} of $\triangle p_i p_{s1} p_{s2}$ (= $\mathbb{T}_i$) configured from three positions occupied by r_i, r_{s1}, and r_{s2}. By the algorithm described above, at time t, r_i in $\mathbb{T}_i(t)$ finds the next target point p_{ti} where the line segment $\overline{p_{ct}p_{ti}}$ is $k_a d_u/\sqrt{3}$ in length and is perpendicular to $\overline{p_{s1}p_{s2}}$. In other words, at $t+1$, the altitude of $\triangle p_{ti}p_{s1}p_{s2}$ is the straight line through p_{ti} and perpendicular to $\overline{p_{s1}p_{s2}}$. Likewise, since r_{s1} and r_{s2} also execute the same algorithm, it is easily seen that p_{ct} in $\mathbb{T}_i(t)$ is the orthocenter in $\mathbb{T}_i(t+1)$ at $t+1$.

In Fig. 3, we denote p_i, p_{s1}, p_{s2}, and p_{ct} for simplicity as A, B, C, and H, respectively. The lengths of lines $\overline{AB}$, $\overline{AC}$, and $\overline{BC}$ are denoted as c, b, and a, respectively. The points P, Q, and R are the foot of the perpendicular from the vertices C, B, and A to the vectors $\overrightarrow{AB}$, $\overrightarrow{AC}$, and $\overrightarrow{BC}$, respectively. Moreover, H is the orthocenter of $\triangle ABC$. Since $\overrightarrow{AB}$ and $\overrightarrow{AC}$ are linearly independent, $\overrightarrow{AH}$ can be defined as

$$\overrightarrow{AH} = x\overrightarrow{AB} + y\overrightarrow{AC}, \tag{1}$$

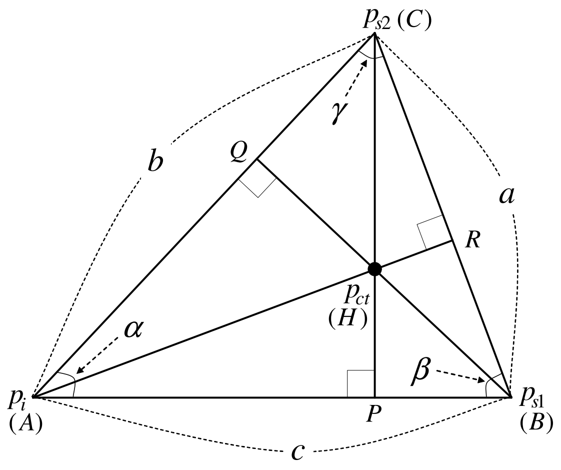


Fig. 3. Convergence of a triangular mesh by the algorithm

where x and y are scaling coefficients, respectively. Since we can easily see that $\overrightarrow{AP} = ((b\cos\alpha)/c)\overrightarrow{AB}$, the following relation holds: $\overrightarrow{PH} = \overrightarrow{AH} - \overrightarrow{AP} = (x - ((b\cos\alpha)/c))\overrightarrow{AB} + y\overrightarrow{AC}$. Thus, the inner product between $\overrightarrow{PH}$ and $\overrightarrow{AB}$ can be expressed as follows:

$$\overrightarrow{PH} \cdot \overrightarrow{AB} = (x - ((b\cos\alpha)/c))c^2 + (bc\cos\alpha)y = 0. \tag{2}$$

Similarly, the following equation holds:

$$\overrightarrow{QH} \cdot \overrightarrow{AC} = (bc\cos\alpha)x + (y - ((c\cos\alpha)/b))b^2 = 0. \tag{3}$$

Now, using (2) and (3), the following simultaneous equations can be obtained:

$$\begin{aligned} cx - (b\cos\alpha)y &= b\cos\alpha \\ (c\cos\alpha)x + by &= c\cos\alpha. \end{aligned} \tag{4}$$

By solving (4), we can obtain the coefficient x as follows:

$$x = \frac{b\cos\alpha(b - c\cos\alpha)}{bc\sin^2\alpha}.$$

Using the cosine formula ($b = c\cos\alpha + a\cos\gamma$), x is expressed as follows: $x = (a\cos\alpha\cos\gamma)/(c\sin^2\alpha)$. In addition, by utilizing the sine formula ($\frac{a}{\sin\alpha} = \frac{c}{\sin\gamma}$), x is rewritten as the following equations: $x = (\cos\alpha\cos\gamma)/(\sin\alpha\sin\gamma)$. Thus, if we do not consider the case of a right triangle, it is straightforward to rewrite x as the following equation:

$$x = \frac{1}{\tan\alpha\tan\gamma} = \frac{\tan\beta}{\tan\alpha\tan\beta\tan\gamma}. \tag{5}$$

Similarly, the coefficient y can be represented as follows:

$$y = \frac{\tan\gamma}{\tan\alpha\tan\beta\tan\gamma}. \tag{6}$$

Using the addition theorems of the trigonometrical function, we can obtain the result of $\tan\alpha\tan\beta\tan\gamma = \tan\alpha + \tan\beta + \tan\gamma$. Thus, (5) and (6) are rewritten as follows:

$$x = \frac{\tan\beta}{\tan\alpha + \tan\beta + \tan\gamma}, y = \frac{\tan\gamma}{\tan\alpha + \tan\beta + \tan\gamma}. \tag{7}$$

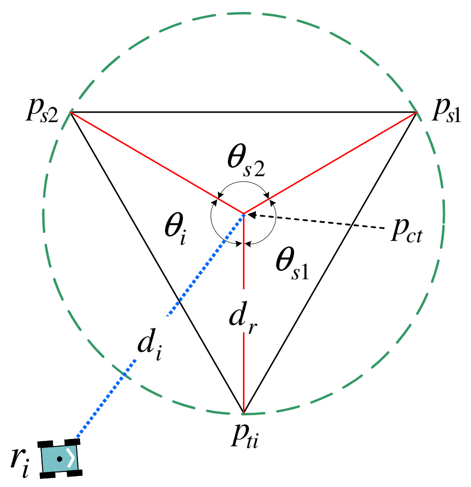


Fig. 4. Motion control of a robot by the algorithm

With respect to a reference point O, (1) can be rewritten as follows: $\overrightarrow{AH} = \overrightarrow{OH} - \overrightarrow{OA} = x(\overrightarrow{OB} - \overrightarrow{OA}) + y(\overrightarrow{OC} - \overrightarrow{OA})$. Now $\overrightarrow{OH}$ can be represented into the following form.

$$\overrightarrow{OH} = (1 - x - y)\overrightarrow{OA} + x\overrightarrow{OB} + y\overrightarrow{OC}. \qquad (8)$$

Substituting (7) into (8), finally, $\overrightarrow{OH}$ is given by

$$\overrightarrow{OH} = \frac{\tan\alpha\overrightarrow{OA} + \tan\beta\overrightarrow{OB} + \tan\gamma\overrightarrow{OC}}{\tan\alpha + \tan\beta + \tan\gamma}. \qquad (9)$$

The equations above include two important properties for our proposed algorithm. From (9), first r_i forms $\mathbb{T}_i(t)$ using p_i and N_i since it can compute p_{ct} and the orthocenter H. Secondly, r_i uses p_{ct} at t as a basis to generate $\mathbb{T}_i(t+1)$ with $k_a d_u/\sqrt{3}$ from p_{ct} to p_{ti} in length at $t+1$. In detail, since r_i has no common coordinates (see Section III) and p_{ct} and H between t and $t+1$ remain unchanged, $\overrightarrow{OH}$ in (9) means the position vector toward p_{ct} with respect to the origin of r_i's local coordinates.

Next, under the adaptive triangular mesh generation algorithm, three neighboring robots attempt to cooperatively configure themselves into an equilateral triangle adapting to the density of contamination. Let us consider the circumscribed circle of a triangle $\triangle p_{ti}p_{s1}p_{s2}$ configured from three positions occupied by r_i, r_{s1}, and r_{s2} where the circumcenter is p_{ct} and the circumradius is $k_a d_u/\sqrt{3}$ in length. From the desired configuration, we design the motion of each robot by controlling the distance d_i from p_{ct} and the internal angle θ_i between $\overline{p_{ct}p_{ti}}$ and $\overline{p_{ct}p_{s2}}$ (see Fig. 4). First, d_i is controlled by the following equation

$$\dot{d}_i(t) = -a(d_i(t) - d_r), \qquad (10)$$

where a is a positive constant and d_r represents $k_a d_u/\sqrt{3}$. Indeed, the solution of (10) is $d_i(t) = |d_i(0)|e^{-at} + d_r$ that converges exponentially to d_r as t approaches infinity. Secondly, θ_i is controlled by the following equation

$$\dot{\theta}_i(t) = k(\theta_{s1}(t) + \theta_{s2}(t) - 2\theta_i(t)), \qquad (11)$$

where k is a positive number. Using the feature of a triangle whose total external angle is 2π, (11) can be rewritten as

$$\dot{\theta}_i(t) = k'(\frac{2}{3}\pi - \theta_i(t)), \qquad (12)$$

where k' is $3k$. Likewise, the solution of (12) is $\theta_i(t) = |\theta_i(0)|e^{-k't} + 2\pi/3$ that converges exponentially to $2\pi/3$ as t approaches infinity. Note that (10) and (12) imply that the trajectory of r_i converges to an equilibrium state $[d_r\ \frac{2}{3}\pi]^T$. From Fig. 4, this also implies that the equilibrium for θ_i is defined as $\theta_i = \theta_{s1}$ since $\triangle p_{ti}p_{ct}p_{s1}$ and $\triangle p_{ti}p_{ct}p_{s2}$ are eventually congruent. In order to show the convergence into the state $[d_i(t)\ \theta_i(t)]^T$, we will take advantage of stability based on Lyapunov's theory [19]. The convergence into the desired configuration is one that minimizes the energy level of a scalar function. Consider the following scalar function:

$$f_i(d_i, \theta_i, \theta_{s1}) = \frac{1}{2}(d_i - d_r)^2 + \frac{1}{2}(\theta_{s1} - \theta_i)^2. \qquad (13)$$

This scalar function is always positive definite except $d_i \neq d_r$ and $\theta_i \neq \theta_{s1}$. The derivative of the scalar function is given by $\dot{f}_i = -(d_i - d_r)^2 - (\theta_{s1} - \theta_i)^2$, which is negative definite. The scalar function is radially unbounded since it tends to infinity as $||[d_i(t)\ \theta_i(t)]^T|| \rightarrow \infty$. Therefore, the equilibrium state is asymptotically stable, implying that r_i reaches a vertex of the desired triangle. Now we show the convergence of the algorithm for n robots. The n-order scalar function $\mathbf{F}$ is defined as

$$\mathbf{F} = \sum_{i=1}^{n} f_i(d_i, \theta_i, \theta_{s1}). \qquad (14)$$

It is straightforward to verify that $\mathbf{F}$ is positive definite and $\dot{\mathbf{F}}$ is negative definite. $\mathbf{F}$ is radially unbounded since it tends to infinity as t approaches infinity. Consequently, n robots move toward the equilibrium state.

V. SIMULATION RESULTS

In this section, we performed simulations of filling an area of interest with a swarm of robots in order to show the validity of our proposed algorithm. In these simulations, we represent the area of varying density as the colored circle that will be sparsely or densely populated with robots.

First, in order to assist in understanding the self-configuration of robot swarms from an initial distribution, Fig. 5 presents snapshots for the simulation result by 100 robots. With initial distributions in Fig. 5-(a), robots configured themselves in the 2-dimensional plane (see Fig. 5-(c)). After constructing an equilateral triangle network, we investigated how they adapt their triangular mesh network according to the assigned densities. The snapshots from Fig. 5-(d) to -(i) show that robots could generate adaptive triangular meshes based on k_a and d_a. If the series of snapshot are carefully observed, the number of robots within the area of interest represented as the red circle increases. In detail, it is initially observed in Fig. 5-(d) that there are 7 robots within the area. By executing the adaptive triangular mesh generation algorithm repeatedly, the number of robots located at the area becomes 17 robots in the final distribution

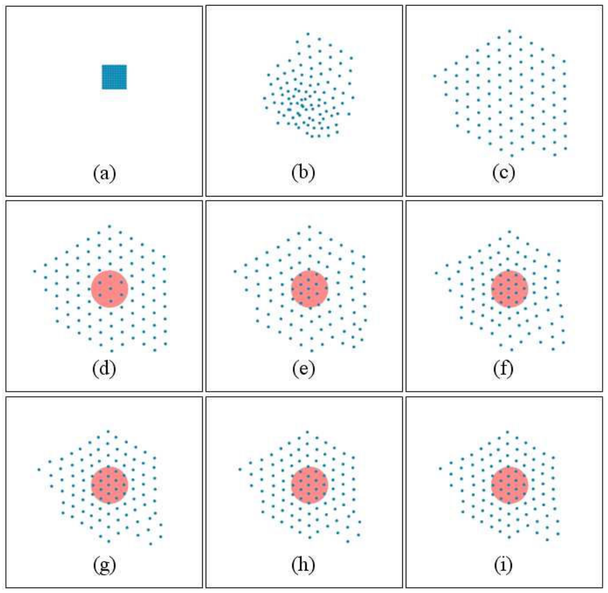
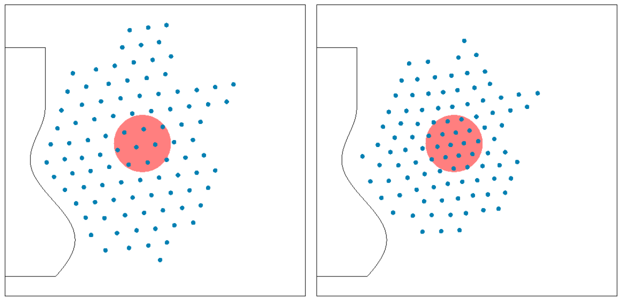


Fig. 5. Simulation result for the whole deployment by 100 robots ((a)∼(c): uniform configuration [16][17], (d)∼(i): adaptive configuration with high center density of 0.7)



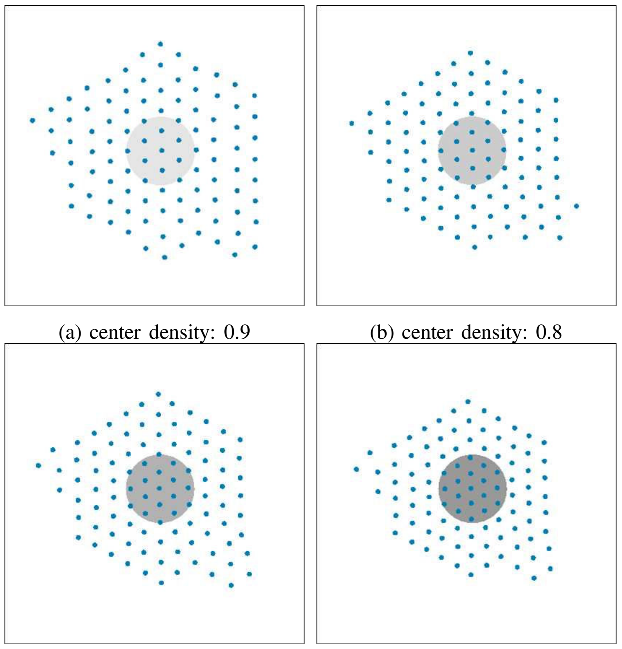
(a) uniform configuration (b) adaptive configuration

Fig. 6. Adaptive triangular mesh generation with geographic borders

(see Fig. 5-(i)). Compared with Fig. 5-(c), the overall size of the final distribution was reduced. Likewise, Fig. 6 shows the result from another simulation when geographical borders exist.

Secondly, Fig. 7 shows the results from simulations with four different degrees of center density. In the figure, the number of robots within the circle increases in proportion to the degree of density. Likewise, the size of swarm in its final converged shape varies according to the center density. Higher densities forced robots decrease the interval between neighboring robots.

Thirdly, we performed simulations for multiple varying densities in a single swarm. Figs. 8-(a) and -(b) show the results of two identical center densities. Although a swarm was spilt into two smaller groups, robots could adaptively configure themselves encompassing the desired area. Figs. 8-(c) and -(d) present the results when the center densities vary: 0.7 on the left hand side and 0.9 on the right hand side. The higher center density area is more densely populated. In Figs. 8-(e) and -(f), the simulation was performed for three different center densities in a swarm: 0.9, 0.7, and 0.8



(a) center density: 0.9 (b) center density: 0.8

(c) center density: 0.7 (d) center density: 0.6

Fig. 7. Simulation results according to varying densities

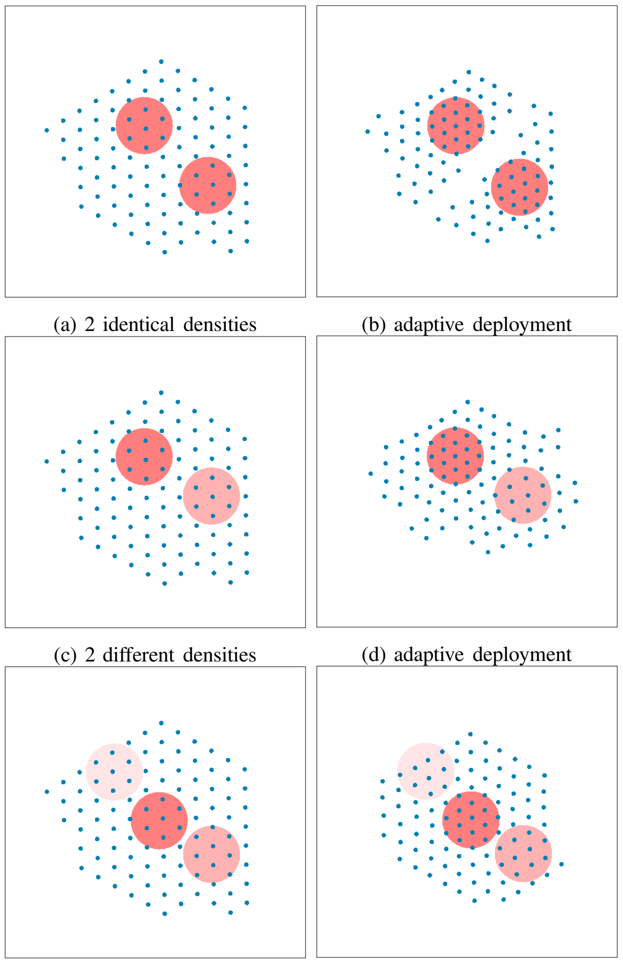
from left to right. Robots could adapt the interval to each other according to the varying densities, which is similar to previous simulations. The overall size of swarm also varies accordingly.

VI. CONCLUSION

The adaptive triangular mesh generation problem was addressed to disperse a swarm of mobile robots adapting to the degree of contamination. There were several major assumptions underlying our proposed approach to this problem: no robot identifiers, no common coordinates or global orientation, and no direct communication. Robots computed their desired position without requiring memories of past actions or states. Under such conditions, the proposed adaptive mesh generation algorithm enables a large-scale swarm of robots to configure themselves into triangular patterns while changing the uniform interval according to the density that can be detected by sensors. We took advantage of the fact that, among all the possible types of n-polygons, the triangle is highly scalable, and less influenced by the number of neighboring robots. To form the desired pattern, robots were allowed to interact with only two selected neighbors at each time. By collecting such local behavior of each robot, a swarm of robots composed of triangular meshes was self-configured into the area of varying degrees of density. The properties of the algorithm was shown mathematically, and also verified through extensive simulations. Finally, we expect that the proposed approach can be used as a simple and effective way to deploy mobile sensor networks for coverage in unknown areas of interest.

REFERENCES

[1] H. Choset, "Coverage for robotics-a survey of recent results," Annals of Math. and Artificial Intelligence, 31:113-126, 2001



(a) 2 identical densities (b) adaptive deployment

(c) 2 different densities (d) adaptive deployment

(e) 3 different densities (f) adaptive deployment

Fig. 8. Simulation results for varying local densities

[2] I. Suzuki and M. Yamashita. "Distributed anonymous mobile robots: formation of geometric patterns," SIAM Jour. Computing, 28(4):1347-1363, 1999

[3] Y. Ikemoto, Y. Hasegawa, T. Fukuda, and K. Matsuda. "Graduated spatial pattern formation of robot group," Information Sciences, 171(4):431-445, 2005

[4] M. Shimizu, T. Mori, and A. Ishiguro. "A development of a modular robot that enables adaptive reconfiguration," IEEE/RSJ Int. Conf. Intelligent Robots and Systems, 174-179, 2006

[5] T. Balch and M. Hybinette. "Social potentials for scalable multi-robot formations," IEEE Int. Conf. Robotics and Automation, 73-80, 2000

[6] A. Howard, M. J. Mataric, and G. S. Sukhatme. "Mobile sensor network deployment using potential fields: a distributed, scalable solution to the area coverage problem," Int. Sym. Distr. Autono. Robotic Systems, 299-308, 2002

[7] W. Spears, D. Spears, J. Hamann, and R. Heil. "Distributed, physics-based control of swarms of vehicles," Autono. Robots, 17(2-3):137-162, 2004

[8] K. Fujibayashi, S. Murata, K. Sugawara, and M. Yamamura. "Self-organizing formation algorithm for active elements," IEEE Sym. Reliable Distr. Systems, 416-421, 2002

[9] J. Reif and H. Wang. "Social potential fields: a distributed behavioral control for autonomous robots," Robotics and Autono. Systems, 27(3):171-194, 1999

[10] Y. F. Zheng and W. Chen, "Mobile robot team forming for crystallization of protein," Autono. Robots, 23(1):69-78, 2007

[11] J. McLurkin and J. Smith. "Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots," Int. Sym. Distr. Autono. Robotic Systems, 831-890, 2004

[12] B. Shucker, T. Murphey, and J. K. Bennett. "A method of cooperative control using occasional non-local interactions," IEEE Int. Conf. Robotics and Automation, 1324-1329, 2006

[13] W. Jatmiko, K. Sekiyama, and T. Fukuda, "A particle swarm-based mobile sensor network for odor source localization in a dynamic environment," M. Gini and R. Voyles (eds.), *Distributed Autonomous Robotic Systems 7*, Springer Japan, 71-80, 2007

[14] B. Jung and G. S. Sukhatme, "Tracking targets using multiple robots: the effect of environment occlusion," Autonomous Robots, 13(3):191-205, 2002

[15] K. N. Krishnanand, P. Amruth, and M. H. Guruprasad, "Glowworm-inspired robot swarm for simultaneous taxis towards multiple radiation sources," IEEE Int. Conf. Robotics and Automation, 958- 963, 2006

[16] G. Lee and N. Y. Chong, "Self-configurable mobile robot swarms with hole repair capability," IEEE/RSJ Int. Conf. Intelligent Robots and Systems, pp.1403-1408, 2008

[17] G. Lee, S. Yoon, and N. Y. Chong, "Deploying real mobile robot swarms equipped with dual rotating infrared sensors," Proc. 5th Int. Conf. Ubiquitous Robots and Ambient Intelligence, pp.306-310, 2008

[18] S. Ghosh, K. Basu, and S. K. Das. "An architecture for next-generation radio access networks," IEEE Network, 19(5):35-42, 2005

[19] J. E. Slotine and W. Li, *Applied nonlinear control*. Prentice-Hall, 1991